TechTarget

Special Report

Cache Memory Basics

Cache memory is fast and it is expensive. Check out this quick guide for an overview on some of the basic concepts surrounding cache memory and best practices for leveraging cache memory technologies.

What is the Difference Between Cache Memory and RAM Cache?

One response to this question might be one of a real estate agent's old favorites: “Location, location, location!” Cache memory is usually part of the central processing unit, or part of a complex that includes the CPU and an adjacent chipset where memory is used to hold data and instructions that are most frequently accessed by an executing program -- usually from RAM-based memory locations. In the classic von Neumann computer, the RAM was the “chalkboard” where processors did the math of a program. Placing this data store closer to the processor itself -- so data requests and responses didn’t have to traverse the motherboard bus -- reduced the wait time or latency associated with processing and delivered better than average latency and faster chip performance.

A RAM cache, by contrast, tends to include some permanent memory embedded on the motherboard and memory modules that can be installed by the consumer into dedicated slots or attachment locations. These memories are accessed via the mainboard bus (channels or conduits etched into the motherboard that interconnect different devices and chipsets). CPU cache memory operates between 10 to 100 times faster than RAM, requiring only a few nanoseconds to respond to the CPU request. RAM cache, of course, is much speedier in its response time than magnetic media, which delivers I/O at rates measured in milliseconds.

It should be noted that somewhat slower flash memory is now being used to provide an additional cache at the magnetic media level -- on disk controllers -- in an effort to change the latency characteristics of disk, especially as disks become more capacious and access to data increases. Considerable ink has been spilled to suggest that flash -- or solid-state disks -- will at some point in the future displace magnetic disks altogether as a production storage medium.

How CPU Caching Speeds Processor Performance

The key processor specs are clock speed, number of cores and sizes of the level 1 and level 2 caches. It's easy to see how clock speed and the number of cores affect CPU performance, but what impact do level 1 and level 2 cache sizes have? Why is CPU caching important?

A CPU typically has a much higher clock speed than the system bus. For example, my computer has a clock speed higher than 3 GHz and a system bus speed of 400 MHz. This means when the CPU needs to read data from the system's RAM, it must conform to the system bus speed, even though that speed is ridiculously slow compared to the CPU's clock speed.

It's bad enough that a CPU has to spend so much time waiting on data to be read from memory, but often the same data gets read multiple times as a program executes. If the CPU did not have a cache, it would have to stop and wait for the data to be read each time it was requested.

A CPU cache places a small amount of memory directly on the CPU. This memory is much faster than the system RAM because it operates at the CPU's speed rather than the system bus speed. The idea behind the cache is that chip makers assume that if data has been requested once, there's a good chance it will be requested again. Placing the data on the cache makes it accessible faster.

The reason for two CPU caches

Why not just create one large cache on a CPU instead of two small ones?

Using two small caches increases performance. The data requested most recently is typically the data most likely to be needed again. Therefore, the CPU will always check the level 1 cache first. If the requested data is found in the level 1 cache, there's no need for the CPU to even bother checking the level 2 cache. This saves the CPU time because it does not have to search through the majority of the cache memory. (What is cache memory? Read a definition here.)

To further explain CPU caching, let's use the analogy of a library. The books in the library represent data. The librarian represents the CPU.

If someone asks the librarian for a particular book, she must get up, go to the shelves, find the requested book and bring it back to the person waiting for it. This takes some time, and there are other people waiting on the librarian (just as there'd be other instructions waiting on the CPU).

After the person reads the book and returns it, the librarian could put it back on the shelves, but instead she decides to put it on a small shelf next to her desk. This small shelf represents a level 1 cache. If another person were to ask for this book, the librarian could just reach over and pull the book off the shelf rather than having to hunt for it on the main shelves.

As more people return books, the shelf next to the librarian fills up. To make room for more books, she moves books from the shelf next to her desk to a larger shelf a few steps away. This shelf represents the level 2 cache. If someone asked the librarian for a book on this shelf, it would take the librarian a bit of time to find it because she'd first check the shelf next to her desk, then check the second shelf. Even so, it would still take her less time to retrieve the book than if the book were located somewhere in the library's main shelves.

As you can see, CPU caching is designed to keep recently (or frequently) requested data in a place where it is easily accessible. This avoids the delay associated with reading data from RAM.

Using RamMap and VMMap Tools to Troubleshoot Windows Memory Issues

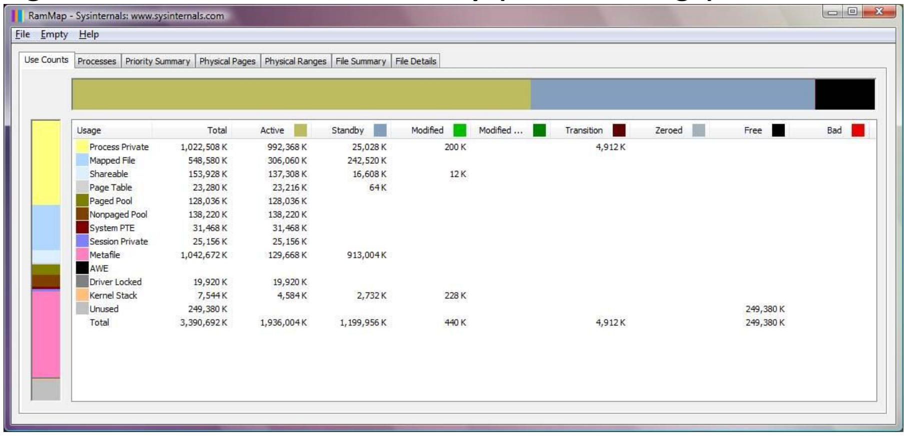
Troubleshooting Windows memory issues requires an in-depth understanding of the operating system along with a working knowledge of how to utilize the Windows Debugger or Performance Monitor. If you're trying to get details such as the kernel stack size or driver memory consumption, you'll need intricate experience with debugger commands and kernel data structures. Even the most veteran system administrator can be challenged when peering into a process address space to determine private versus shared memory utilization or heap size.

Fear not: RamMap and VMMap make troubleshooting memory issues easy. You can download these free tools from the Sysinternals website. Both tools were written by Mark Russinovich (writer of the Process Explorer and Notmyfault tools, and author of *Windows Internals*) and Bryce Cogswell.

RamMap
RamMap is used to display system and process memory statistics and utilization. It provides a summary tab called "Use Counts," which lists all the various system memory regions such as paged and nonpaged pool, process private, shareable, driver space, kernel stack, and mapped files. It also displays the amount of cache file space referred to as Metafile.

All of these regions are then further categorized into the different types of physical memory consumption such as active, standby, modified, transition, zeroed, free or bad. Each of these columns can be sorted by clicking on the column header. All of these terms are explained in Russinovich's *Windows Internals* book. As you can see in Figure 1, the data is neatly presented in a graphical, tabular view:

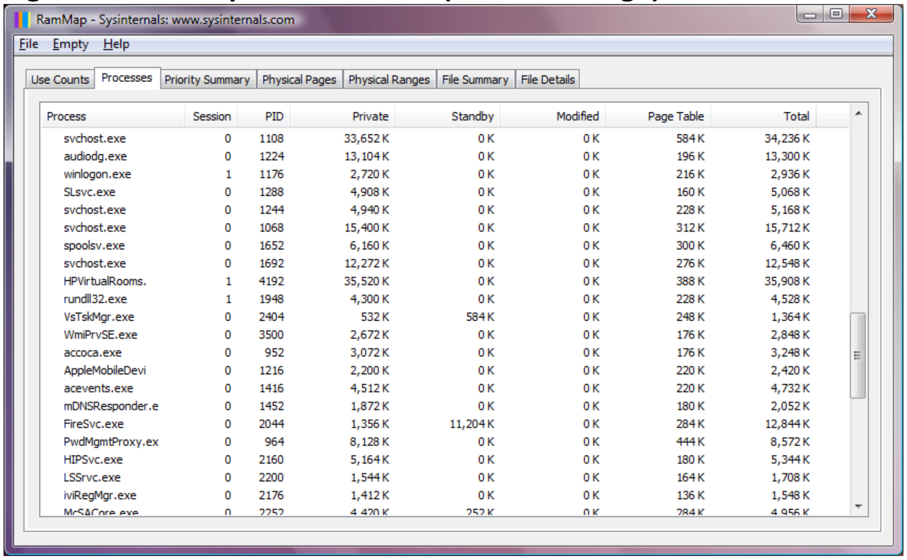
Figure 1: Use Count data in RamMap (click to enlarge)



Usage	Total	Active	Standby	Modified	Modified ...	Transition	Zeroed	Free	Bad
Process Private	1,022,508 K	992,368 K	25,028 K	200 K		4,912 K			
Mapped File	548,580 K	306,060 K	242,520 K						
Shareable	153,928 K	137,308 K	16,608 K	12 K					
Page Table	23,280 K	23,216 K	64 K						
Paged Pool	128,036 K	128,036 K							
Nonpaged Pool	138,220 K	138,220 K							
System PTE	31,468 K	31,468 K							
Session Private	25,156 K	25,156 K							
Metafile	1,042,672 K	129,668 K	913,004 K						
AWE									
Driver Locked	19,920 K	19,920 K							
Kernel Stack	7,544 K	4,584 K	2,732 K	228 K					
Unused	249,380 K							249,380 K	
Total	3,390,692 K	1,936,004 K	1,199,956 K	440 K		4,912 K		249,380 K	

RamMap also reveals process memory utilization on the "Processes" tab. Here you will find all the processes listed, along with their corresponding private memory utilization. The data also includes any process memory that is occupying the standby or modified page list, and the amount of memory used for page table entries.

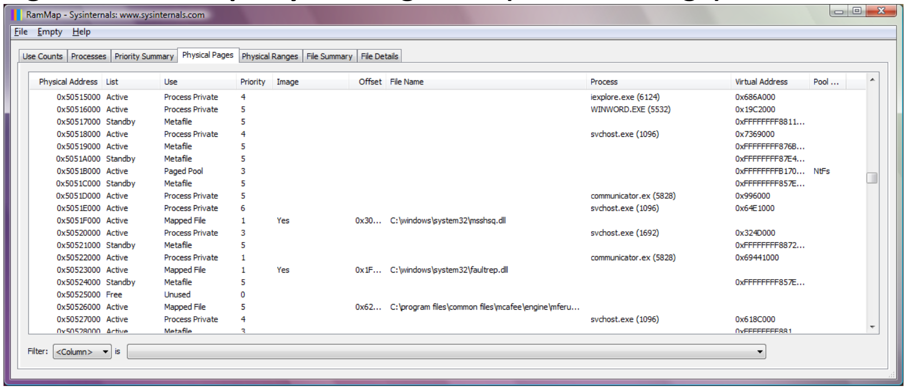
Figure 2: RamMap Processes tab (click to enlarge)



Process	Session	PID	Private	Standby	Modified	Page Table	Total
svchost.exe	0	1108	33,652 K	0 K	0 K	584 K	34,236 K
audiodg.exe	0	1224	13,104 K	0 K	0 K	196 K	13,300 K
winlogon.exe	1	1176	2,720 K	0 K	0 K	216 K	2,936 K
SLsvc.exe	0	1288	4,908 K	0 K	0 K	160 K	5,068 K
svchost.exe	0	1244	4,940 K	0 K	0 K	228 K	5,168 K
svchost.exe	0	1068	15,400 K	0 K	0 K	312 K	15,712 K
spoolsv.exe	0	1652	6,160 K	0 K	0 K	300 K	6,460 K
svchost.exe	0	1692	12,272 K	0 K	0 K	276 K	12,548 K
HPVirtualRooms.	1	4192	35,520 K	0 K	0 K	388 K	35,908 K
rundll32.exe	1	1948	4,300 K	0 K	0 K	228 K	4,528 K
VsTskMgr.exe	0	2404	532 K	584 K	0 K	248 K	1,364 K
WmiPrvSE.exe	0	3500	2,672 K	0 K	0 K	176 K	2,848 K
accoca.exe	0	952	3,072 K	0 K	0 K	176 K	3,248 K
AppleMobileDevi	0	1216	2,200 K	0 K	0 K	220 K	2,420 K
acevents.exe	0	1416	4,512 K	0 K	0 K	220 K	4,732 K
mDNSResponder.e	0	1452	1,872 K	0 K	0 K	180 K	2,052 K
FireSvc.exe	0	2044	1,356 K	11,204 K	0 K	284 K	12,844 K
PwdMgmtProxy.ex	0	964	8,128 K	0 K	0 K	444 K	8,572 K
HIPSvc.exe	0	2160	5,164 K	0 K	0 K	180 K	5,344 K
LSSrvc.exe	0	2200	1,544 K	0 K	0 K	164 K	1,708 K
iviRegMgr.exe	0	2176	1,412 K	0 K	0 K	136 K	1,548 K
McSACore.exe	0	2252	4,420 K	252 K	0 K	284 K	4,956 K

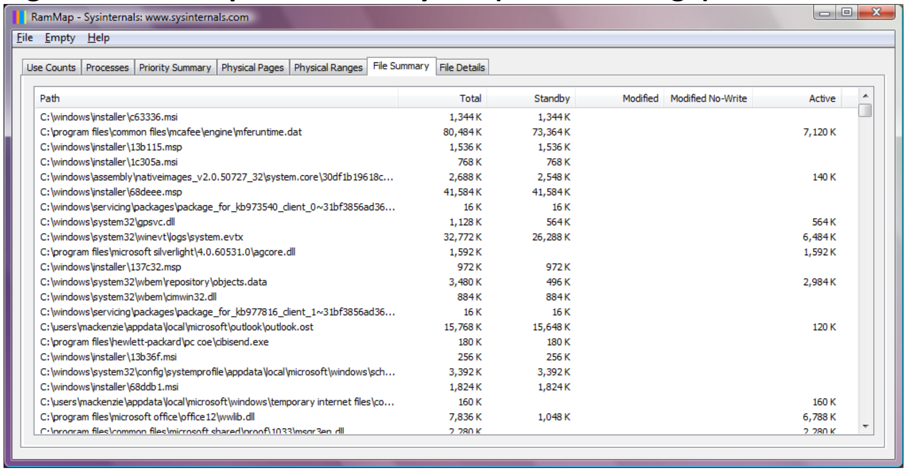
Another use for RamMap is to display the actual, physical memory usage, page by page, identifying attributes such as the memory list, use, filename, process, virtual address and pool tags. Each of these columns can be sorted and there is a filter feature which allows you to selectively analyze the data.

Figure 3: RamMap Physical Pages tab (click to enlarge)



Finally, RamMap does an excellent job of revealing cached file activity and data. You can use the "File Summary" and "File Details" tabs to drill down on the system file cache to determine what is occupying the space. As seen in Figure 4, you can determine the file path, the size it is occupying, and whether the corresponding memory is on the active, standby, or modified page list.

Figure 4: RamMap File Summary tab (click to enlarge)

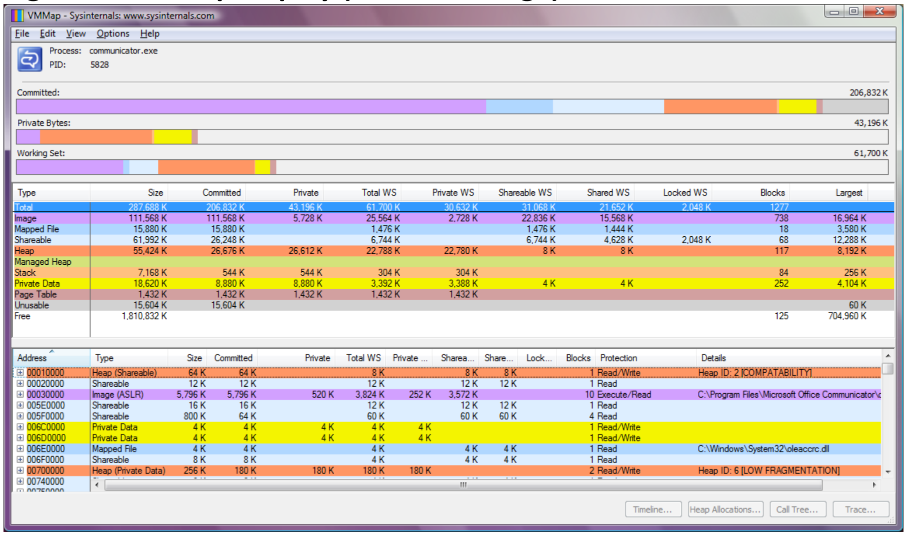


VMMap

Up until now, we have seen how RamMap can reveal system and process memory usage. If the memory issue you are troubleshooting appears to be related to a particular process or application, it may be necessary to take a closer look by using VMMap. VMMap is a process-oriented tool that allows you to view an existing process or trace a new one and observe its memory usage in far greater detail than RamMap.

When VMMap launches, it prompts you to select an existing process you want to investigate, or to start a new one. If you launch a new process, you will be able to trace memory utilization such as heap and virtual allocations. In Figure 5 below, I selected the communicator.exe process.

Figure 5: VMMap display (click to enlarge)



Once the main window of VMMap is displayed, you can see the screen is divided into sections. The top section is a graphical summary view of the process memory consumption. It is divided into committed space, private bytes, and the working set. In the middle portion of the screen, the memory utilization is categorized in terms of use such as private data, shareable, image, mapped files and heap size. Finally, the lower portion of the screen reveals for each virtual address what the corresponding page type is, the size, amount of working set used, page protection, any blocks, and details of the region. The color coding allows you to quickly see how much space a particular type of memory is consuming.

VMMap provides two additional views of the process address space including a "strings" view and a "fragmentation" view. The strings view allows you to search for any character readable strings that are present in the address space. The fragmentation view displays the process virtual address space in a color-coded fashion so you can see the various allocations, their size, and how contiguous they are.

TechTarget

Free resources for technology professionals

TechTarget publishes targeted technology media that address your need for information and resources for researching products, developing strategy and making cost-effective purchase decisions. Our network of technology-specific Web sites gives you access to industry experts, independent content and analysis and the Web's largest library of vendor-provided white papers, webcasts, podcasts, videos, virtual trade shows, research reports and more —drawing on the rich R&D resources of technology providers to address market trends, challenges and solutions. Our live events and virtual seminars give you access to vendor neutral, expert commentary and advice on the issues and challenges you face daily. Our social community IT Knowledge Exchange allows you to share real world information in real time with peers and experts.

What makes TechTarget unique?

TechTarget is squarely focused on the enterprise IT space. Our team of editors and network of industry experts provide the richest, most relevant content to IT professionals and management. We leverage the immediacy of the Web, the networking and face-to-face opportunities of events and virtual events, and the ability to interact with peers—all to create compelling and actionable information for enterprise IT professionals across all industries and markets.

Related TechTarget Websites

- SearchDataBackup
- SearchCloudStorage
- SearchDisasterRecovery
- SearchVirtualStorage
- SearchSMBStorage
- SearchStorage